RESEARCH PLAN PROPOSAL

DESIGN OF NEW NEURO-CRYPTOGRAPHY ALGORITHM FOR BETTER AND SECURED KEY EXCHANGE PROCESS

For registration to the degree of
Doctor of Philosophy

IN THE FACULTY OF SCIENCE

THE IIS UNIVERSITY, JAIPUR

Submitted by

Deepshikha Sharma
Enroll. No. – ICG/2013/16425

Under the Supervision of

Dr. Amita Sharma
Assistant Professor,
THE IIS UNIVERSITY,
Jaipur (Raj).

Department Of Computer Science

May 2014

CONTENTS

TITLE	PAGE NUMBER
1. Topic	3
2. Introduction	3
2.1. Classical Cryptography	3
2.1.1. Symmetric key Cryptography	4
2.1.2. Public key Cryptography	5
2.2. Quantum Cryptography	5
2.3. Neural Cryptography	6
2.3.1. Artificial Neural Network	7
2.3.2. Neural Network Models	8
2.3.3. Activation Function	8
2.3.4. Learning Rules	9
2.3.5. Tree Parity Machine Architecture	12
2.3.6. Neural key Exchange Algorithm	13
2.3.7. Attacks on Neural Key Exchange Algorithm	15
2.3.8. Using Neural Cryptography with Classical Cryptography	17
2.3.9. Other Proposed Neural Key Exchange Algorithms	17
2.3.10. Neural Network Tools	18
3. Literature Review	20
4. Motivation/Justification and Relevance	22
5. Objectives	23
6. Plan of Work and Methodology	23
6.1. Methodology	23
6.2. Work Plan	24
7. Expected Outcome of the Proposed Research Work	25
8. Place of Work and Facilities Available	25
9. References	25

1. TOPIC

Design of new neuro-cryptography algorithm for better and secured key exchange process.

2. INTRODUCTION

Secure communication of messages has been a goal of all civilizations since ages. People have been trying to keep secrets for as long as there have been secrets, and people. Scytales and ciphers were among the most popular methods of encoding information in ancient Greece. A Scytale is a tool used to perform a transposition coded text, consisting of a cylinder with a strip of parchment wound around it on which is written a message. Ancient ciphers, meanwhile, matched numbers and letters in different combinations to form codes which could be translated only by people with a specific key. Cryptography has evolved over the years to a much more advanced state that our brains alone are incapable of breaking [16].

Cryptography manages secret knowledge by taking a message in plaintext and converting it to an unintelligible message to any prying eyes. The need for cryptography is growing exponentially, there are a number of security services that should be addressed by cryptography, with the most important being confidentiality, authenticity, and accountability. Confidentiality protects against unauthorized release of the message, authenticity assures the recipient of the message that the sender is who they say they are, and accountability ensures the sender of the message is accountable for the contents. These goals are achieved by various cryptography approaches [16].

This review document gives an overview of the cryptography techniques and in this document we also discuss neural network technique in detail, and how it can be applied to cryptography for secure message transfer.

2.1 CLASSICAL CRYPTOGRAPHY

Cryptography is the practice and study of hiding information. It is the most useful aspect for secure communication. Cryptography not only protects data from theft or modification but also used for user certification. Cryptography can also be defined as the conversion of data into a cipher code that can be decoded and sent through a public or private network [4].

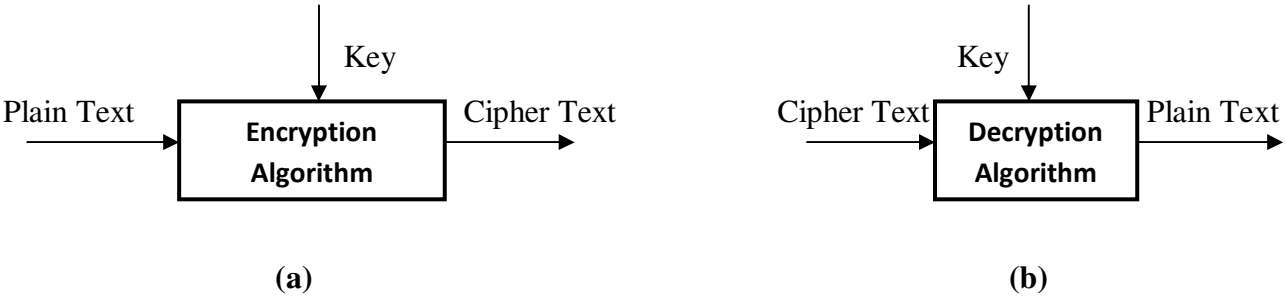


Fig.1. Cryptography Process

The detailed operation of encryption and decryption is controlled both by algorithm and in each instance by a secret parameter called key [4].

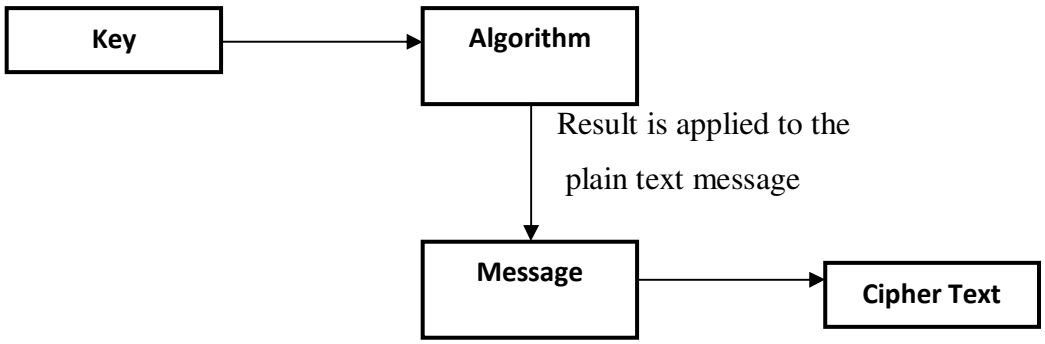


Fig.2. Use of a key

If an opponent (attacker) captures a message as it passes between two people, he can view the message, but it appears in its encrypted form and is therefore unusable. Even if this attacker knows the algorithm that the two people are using to encode and decode their information, without the key this information remains useless to him. Hence key is very important. Based on the key the classical cryptography is divided into two broad areas- Symmetric and Public-key cryptography [4].

2.1.1. Symmetric-key Cryptography

Symmetric key encryption uses the same cryptographic algorithm and the same key to encrypt and decrypt the messages. The key is chosen pseudo-randomly from a subset of all possible key values. A disadvantage is the key management necessary to use them securely. Each distinct pair of communicants must share a different key and each cipher text exchanged as well. The number

of keys required increases as the square of the numbers, of network members, which requires complex key management schemes to keep them all secret.

Examples of commonly used symmetric key encryption algorithms are Data Encryption Standard (DES), 3DES, Rivest Cipher (RC-4), and International Data Encryption Algorithm (IDEA).

2.1.2. Public-key (Asymmetric) Cryptography

Public key cryptography is also known as asymmetric key encryption. As the name indicates, it requires two different but mathematically related keys, one to encrypt a message and the other corresponding key to decrypt the message. Since one key is known publicly, it is called the public key .The other key has to be kept private with one or the other party for the secure communication. It is therefore referred to as the private key.

Though the system avoids the key distribution problem, unfortunately their security depends on unproven mathematical assumptions about the intrinsic difficulty of certain operations. The most popular key cryptosystem RSA (Rivest-Shamin-Adleman) gets its security from the difficulty of factoring large numbers. This means that if ever mathematicians or computer scientists come up with fast and clever procedures for factoring large numbers, the whole privacy and discretion of widespread cryptosystems could vanish overnight.

2.2 Quantum Cryptography

The development of quantum cryptography was motivated by the short-comings of classical cryptographic methods; it is an approach of securing communications by applying the phenomenon of quantum physics. Quantum cryptography relies on two important elements of quantum mechanics- the Heisenberg Uncertainty principle and the principle of photon polarization. The Heisenberg Uncertainty principle states that, it is not possible to measure the quantum state of any system without disturbing that system. The principle of photon polarization states that, an eavesdropper cannot copy unknown qubits i.e. unknown quantum states, due to no cloning theorem [2].

It is impossible to establish a secret key with conventional communications, and so key distribution has relied on the establishment of a physically secure channel or the conditional security of “difficult” mathematical problems in public-key cryptography. However, provably secure key distribution becomes possible with quantum communications. In this process only the

key is distributed over the quantum channel and not the encrypted message. That is why we need two channels between sender and receiver. One public channel for transmission of encrypted message and another channel which is called quantum channel, used for key distribution. Hence, a more accurate name is quantum key distribution (QKD) [2].

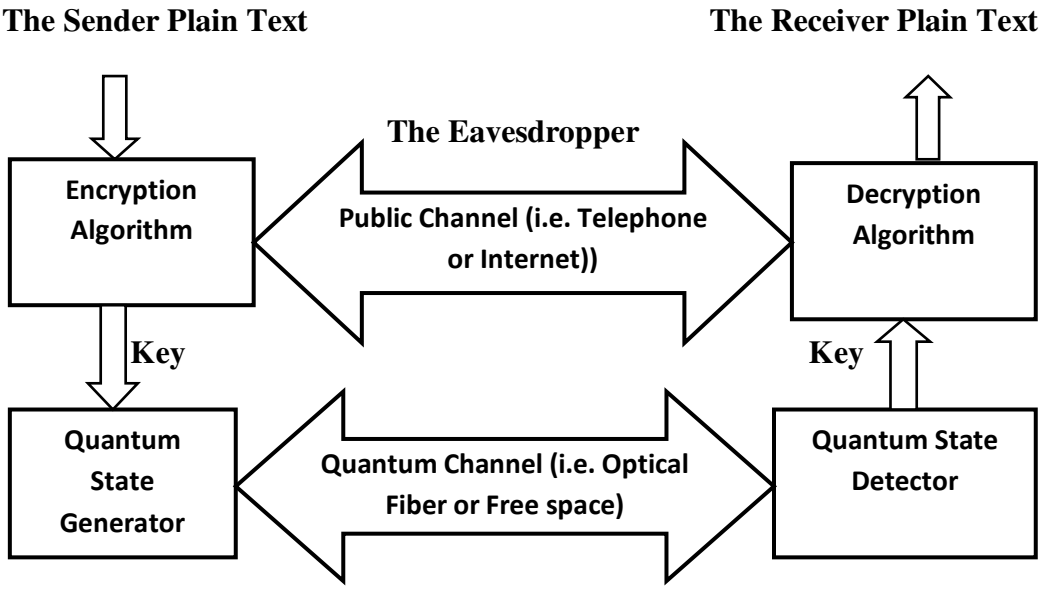


Fig.3. A Quantum Cryptographic communication system for securely transferring random key.

The laws of quantum physics guarantee the security of quantum key distribution systems in theory, but do not guarantee the security of implementing such a system. A few of the problems allied with quantum key distribution completely replacing classical key distribution are as follows: high bit errors during the transfer of the quantum bits, no quantum repeater available so the distance over which communication can happen is too short for our globalized world, and finally sending one photon at a time is a challenge and if it is not met, the security of the system is compromised [16]. To overcome the limitations of quantum cryptography a new cryptographic techniques called neural cryptography was ivolved.

2.3 Neural Cryptography

Neural cryptography is a new branch of cryptography which incorporates artificial neural networks with cryptography. It is dedicated to analyzing the application of stochastic algorithm namely, neural network algorithm, for use in either cryptanalysis or encryption. The first work

that is known on this topic can be traced back to 1995 in an IT Master Thesis by Sebastien Dourlens. Due to its recent development, there hasn't been much practical applications as of yet.

2.3.1. Artificial Neural Network (Neural Network/ANN)

An artificial neural network is an information processing example that is stimulated by the way biological nervous system, such as the brain, process information. The key element of this paradigm is the structure of the information processing system. It is composed of a large number of highly interconnected processing elements (neurons) working in accord to solve specific problems. ANNs, like people, learn by example. An ANN is configured for a specific application, such as pattern recognition or data classification, through a learning process. Learning in biological system involves adjustments to the synaptic connections that exist between the neurons. This is true for ANNs as well [19].

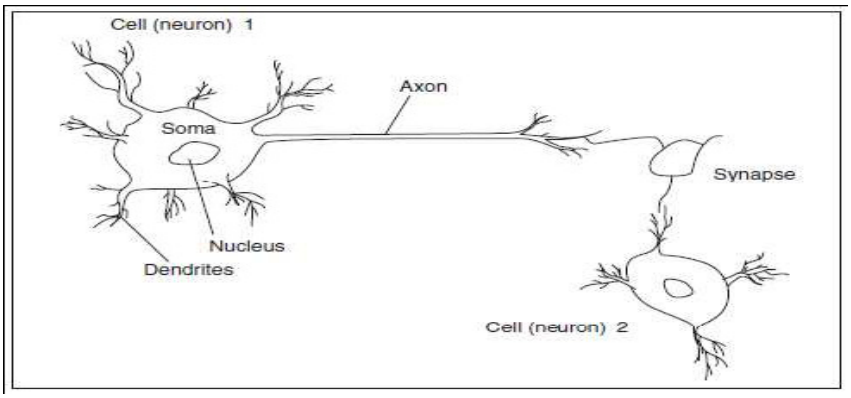


Fig.4. Two interconnected biological cells.

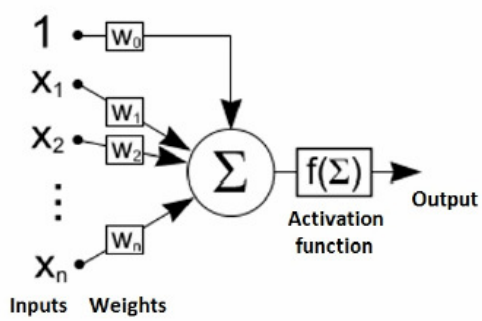


Fig.5. A simple artificial neuron.

Artificial neural networks are embedded in the form of tree parity machines (TPM), which are used for secret key generation through synchronization. For this purpose both the sender and receiver need TPMS.

Neural Networks essentially comprise three pieces:

a) The architecture or model.

b) The activation functions.

c) The learning algorithm.

2.3.2. Neural network models

The neural network model can be broadly divided into the following three types:

(a) Feed-forward networks*:* Each neuron in one layer is connected to every neuron on the next layer. The information is constantly "fed forward " from one layer to the next, also there is no connection among neurons in the same layer.

***(*b) Feedback network**: Unlike feed forward, feedback neural network allows signals to travel in both directions by introducing loops. Such networks are very powerful and complicated.

(c) Self-organization networks: it regards adaptive resonance theory (ART) model and Kohonen model as representatives, and mainly used for cluster analysis.

2.3.3. Activation Function

Activation functions are basically the transfer function which is output from a artificial neuron and it sends signals to the other artificial neuron. There are four forms of activation functions.

a) Threshold

b) Piecewise Linear

c) Sigmoid

d) Guassian

All activation functions are different from each other [3].

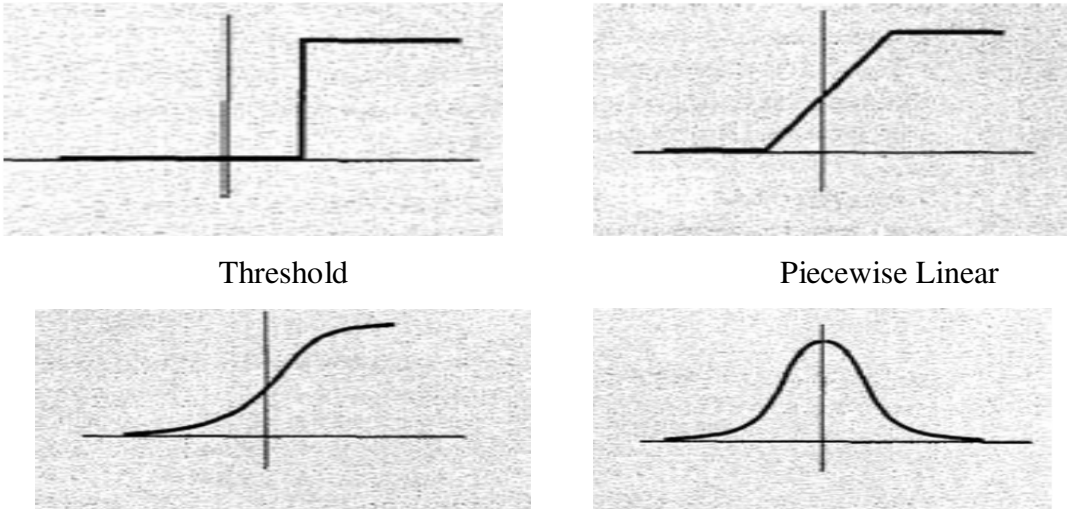


Fig.6. Types of activation functions

2.3.4.Learning Rules

a) Hebbian learning rule:

The Hebb learning rule was proposed in 1949, it is the oldest and well-known learning rules. It is an unsupervised learning rule. In this, the learning process of neurons is based on assumption i.e. if the two neighboring neurons activate and deactivate simultaneously then the weights of these connecting neurons should increase otherwise if they act in opposite stages then the weights between them should decrease.

If there is no association, the weight should remain unaffected. This assumption can be described by the formula

$$\Delta w_{ij}=cx_i o_j$$

where w_{ij} is the weight from *i-th* to *j-th* neuron, c is learning constant, x_j is the signal on the *i-th* input and o_j is the output signal.

The training process starts usually with values of all weights set to zero. The absolute values of the weights are usually proportional to the learning time, which is an undesired effect [7].

b) Correlation learning rule:

In this learning rule the assumption made are based on similar assumption as that in hebb learning rule. The weights of the simultaneously responding neurons should be largely positive and the weights of the neurons responding in opposite phase should be largely negative. Mathematically, this can be written that weights should be proportional to the product of states of connected neurons.

$$\Delta w_{ij}=cx_i d_j$$

It is a supervised training method. This training algorithm starts usually with initialization of weights to zero values [7].

c) Instar learning rule:

If input vectors, and weights, are normalized, or they have only binary bipolar values (-1 or +1), then the *net* value will have the largest positive value when the weights have the same values as the input signals. Therefore, weights should be changed only if they are different from signals.

$$\Delta w_i = c(x_i - w_i)$$

Note, that the information required for the weight is only taken from the input signals. This is a very local and unsupervised learning algorithm [7].

d) WTA- Winner Takes All:

The WTA was proposed in 1982 by Kohonen. It is an unsupervised algorithm and widely known. The WTA is a modification of the instar algorithm where weights connecting to the neurons with highest net value are changed. Weights of remaining neurons are left unchanged. Sometimes this algorithm is modified in such a way that a few neurons with the highest *net* values are modified at the same time [7].

e) Outstar learning rule:

Both instar and outstar learning rules were developed by Grossberg (1974). In outstar learning rule it is required that weights connected to the certain node should be equal to the desired outputs for the neurons connected through those weights.

$$\Delta w_{ij} = c(d_j - w_{ij})$$

where d_j is the desired neuron output and c is small learning constant which further decreases during the learning procedure. This is the supervised training procedure because desired outputs must be known [7].

f) Perceptron learning rule:

A perceptron is a single layer neural network with one output unit. The perceptron training rule can be defined for each weight as :

$$w_i = w_i + \Delta w_i$$

$$\text{where, } \Delta w_i = \eta (t - o) x_i$$

where, t is the target output, o is the output of the perceptron, and η is the learning rate .This scenario assumes that we know what the target outputs are supposed to be like. The perceptron training rule converges after a finite number of iterations [7].

g) Widrow- Hoff (LMS) learning rule:

Widrow and Hoff (1962) developed a supervised training algorithm which allows to train a neuron for the desired response. This rule was developed so the square of the difference between *net* and output value is minimized.

$$Error_j = \sum_{p=1}^{P} (o_{jp} - d_{jp})^2$$

where $Error_j$ is the error for *j-th* neuron, p is the number of applied patterns, d_{jp} is the output *for j-th* neuron when *p-th* pattern is applied. This rule is also known as the LMS (Least Mean Square) rule. The LMS rule works well for all type of activation functions. This rule tries to enforce the *net* value to equal to desired value. It is usually not important what the *net* value is , but it is important if the *net* value is positive or negative [7].

h) Linear regression:

The LMS learning rule requires hundreds or thousands of iterations before it converges to the proper solution. Using the linear regression the same result can be obtained in only one step. This method, similar to the LMS rule, assumes a linear activation function, so the *net* values should be equal to desired output values [7].

i) Delta learning rule:

The delta rule overcomes the shortcomings of the perceptron training rule not being guaranteed to converge if the examples are not linearly separable. Delta rule is based on gradient descent search. The delta rule for individual weights can now be written as

$$w_i = w_i + \Delta w_i$$

$$\Delta w_i = \eta \sum_{d \in D} (t_d - o_d) x_{id}$$

The gradient descent algorithm: picks initial random weights, computes the outputs, updates each weight by adding Δw_i and repeats until converges. It will find the global minimum, provided that the learning rate is small enough. If the learning rate is too large, this algorithm runs into the risk of overstepping the global minimum. It's a common strategy to gradually decrease the learning rate. This algorithm works also in case the training examples are not linearly separable [7].

j) Error backpropagation learning rule:

The delta learning rule can be generalized for multilayer networks. Using the similar approach, the gradient of the global error can be computed in respect to each weight in the network [7].

2.3.5. Tree Parity Machine (TPM) Architecture

Neural networks are used for key generation process in neural cryptography and for this tree parity machines are use, which is needed by both the communicating parties. The tree parity machine is a special type of multi-layer feed-forward neural network. It consists of one output neuron, K hidden neurons and K*N input neurons. Inputs to the network are binary.

$x_{ij} \in \{-1, +1\}$

Weights of input and hidden neuron take the values.

$w_{ij} \in \{-L, \ldots, 0, \ldots, +L\}$

Output of each hidden neuron is calculated as a sum of all products of input neurons and its weights.

$$\sigma_i = sgn\left(\sum_{j=1}^{N} w_{ij}\ x_{ij}\right)$$

Signum is a simple function, which returns -1, 0 or 1.

$$sgn(x) = \begin{cases} -1 & if\ x < 0 \\ 0 & if\ x = 0 \\ 1 & if\ x > 0 \end{cases}$$

If the scalar product is 0, the output of hidden neuron is mapped to -1 to ensure a binary output value. The output of neural network is then computed as the product of all values produced by hidden element.

$$\tau = \prod_{i=1}^{K} \sigma_i$$

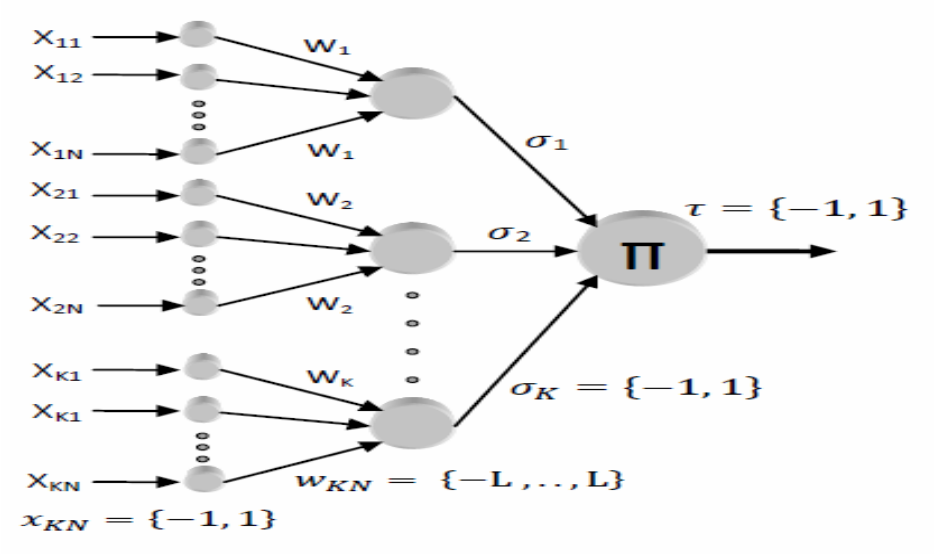


Fig.7. Architecture of Tree Parity Machine

Summarizing, the output of the TPM is in binary and can be written as:

$$\tau = \prod_{i=1}^{K} sgn\left(\sum_{j=1}^{N} w_{ij}\ x_{ij}\right)$$

This tree parity machine plays an important role in the secret key generation process through neural networks.

2.3.6. Neural Key Exchange Algorithm

Neural key exchange process is based on synchronization of two TPMs, each associated with parties wishing to share a secret. We follow the following steps for neural key generation which is based on synchronization of tree parity machines of the sender and receiver:

a) First of determine the neural network parameters i.e.: k, the number of hidden layer units n, the input layer units for each hidden layer unit l, the range of synaptic weight values is done by the two machines A and B.

b) The network weights to be initialized randomly.

c) Repeat d) and e) until synchronization occurs.

d) The inputs of the hidden units are calculated.

e) The output bit is generated and exchanged between the two machines A and B.

f) If the output vectors of both the machines are same i.e. $\tau^{A} = \tau^{B}$ then the corresponding weights are modified using the Hebbian learning rule, Anti-Hebbian learning rule and Random –walk learning rule.

g) After complete synchronization, the synaptic weights are same for both the networks. And these weights are used as secret key [5].

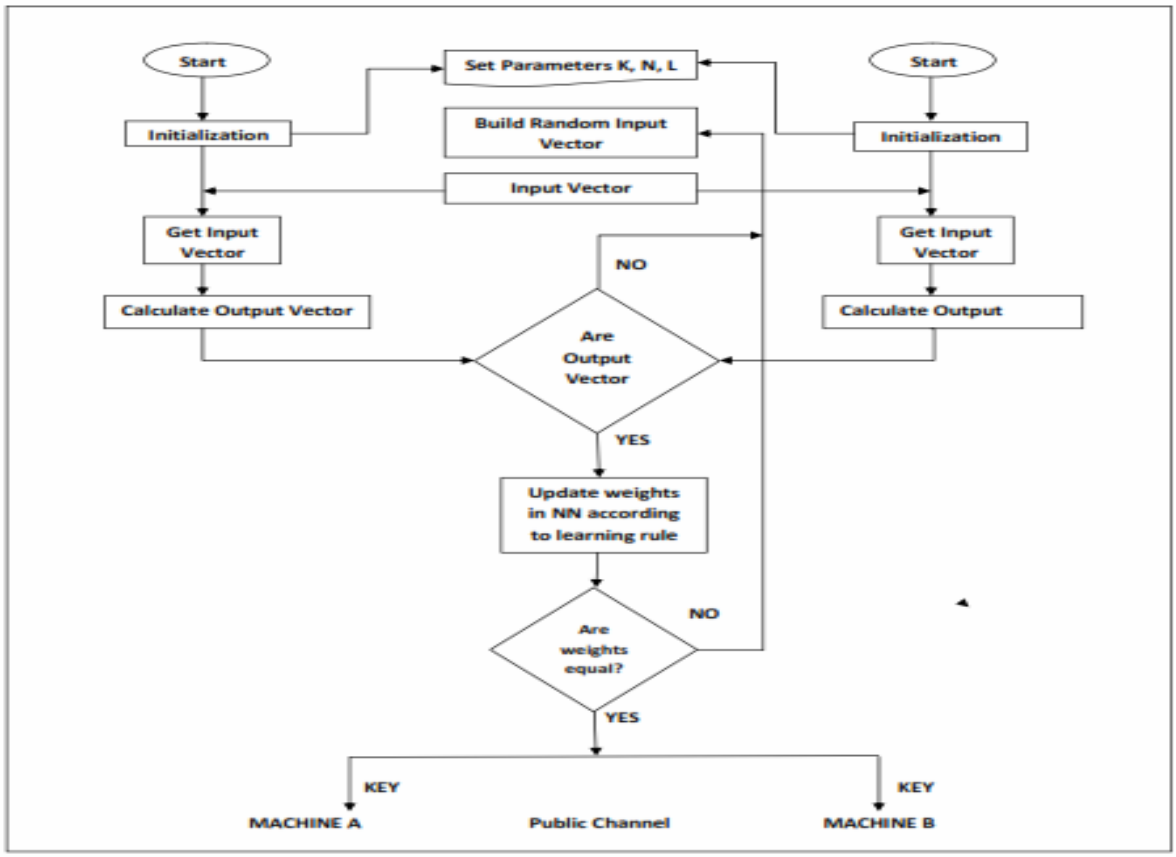


Fig.8. Neural Key Exchange Algorithm

The neural key exchange algorithm is an application of synchronization. The sender and receiver use the TPM with same structure and achieve synchronization between the machines where the number of hidden and input neurons and the synaptic weight range are public parameters whereas the initial randomly chosen weight and internal representation of the TPM are kept private. After full synchronization the modified weights are used as common secret key. Since the internal representation of the communicating parties is not known to the attacker which creates him the difficulty to eavesdrop. But there is the risk of other attack methods that could break the security of the neural cryptography [5].

2.3.7. Attacks on neural key exchange algorithm

The basic attack strategies of neural cryptography are given below:

a) **Simple Attack**:

It is similar to the brute force attack. Here the attacker E's NN has the same structure as that of sender (A) and receiver (B). All that E has to do is to start with random initial weights and to train with the same inputs transmitted between A and B over public channel. Then, the attacker E learns the mutual output bits

$\tau^{A/B}$ between them and applies the same learning rule by replacing τ^{E} with $\tau^{A/B}$

$$\mathbf{w}_k^E = \mathbf{w}_k^E - \tau^A \mathbf{x}_k \Theta \left(\sigma_k^E \tau^{A/B}\right)\left(\tau^A \tau^B\right).$$

b) **Geometric Attack:**

A geometric attack can outperform a simple attack because, in addition to applying the same learning process, E can exploit τ^{E} and the local fields of its hidden units, $h_1^E, h_2^E, \ldots, h_k^E$. When $\tau^{A/B} = \tau^{E}$, E applies only the same learning rule used by A and B . If $\tau^{A/B} \neq \tau^{E}$, then E tries to correct its internal representation of its TPM, $h_1^E, h_2^E, \ldots, h_k^E$. The lower the absolute value of $|h_i^E|$, the higher the probability that $\sigma^{A/B} \neq \sigma^{E}$ and this probability is known as the maximum error prediction ϵ_i^p . Therefore, E will manipulate the hidden unit with minimum $|h_i^E|$ before applying the learning rule. Since the geometric follows a probabilistic behavior, it is not guaranteed that it will succeed in correcting the internal representation of the attacker's TPM.

c) **Majority Attack:**

The majority attack is similar to the geometric attack. However, E can increase the probability to predict the internal representation of any of the partners' TPM. The majority attack is also called as "cooperative attack" since the *M* TPMs are working in a cooperating group rather than as individuals. Instead of using only one TPM, the attacker starts with *M* TPMs with random weight vectors to achieve a zero overlap between them. If $\tau^{A} \neq \tau^{B}$, weights are not updated . But for $\tau^{A} = \tau^{B}$, the attacker must update its own TPM . Then it calculates the output bit of all its attacking networks. If the *m*th attacking network's output bit $\tau^{E,m}$ is not equal to τ^{A}, then the geometric attack is applied individually for every *m*th network and no update will occur before cooperating with other networks. Next, E searches for the common internal representation

among the *M* TPMs and updates their weights according to this majority vote. In order to avoid the expected high correlation between the *M* networks, the majority attack and the geometric attack are applied alternately in even and odd steps, respectively. Also, the majority attack is replaced with the geometric attacks in the first 100 steps.

d) **Genetic Attack:**

In the genetic attack, the attacker starts with only one TPM but is permitted to use *M* TPMs. For a specific value of $\tau^{A/B}$, there are 2^{K-1} different internal representation (using a number of *K* binary variables $\{\sigma_i\}_{i=1}^{K}$) to reproduce this value. The genetic attack handles all these possibilities in a parallel fashion. The genetic attack proceeds as follows:

1) If $\tau^{A} = \tau^{B}$ and E has at most $M/2^{K-1}$ TPMs, 2^{K-1} TPMs are generated and each weights based on one of the possible internal representations. It is called the mutation step.

2) If E has more than $M/2^{K-1}$ TPMs, the mutation step will be essentially an overhead due to the exponential storage needed. The attacker must discard some of the TPMs to avoid exponential storage increase. As a genetic algorithm, the discarding procedure is based on removing the TPMs with the least fittest function. The algorithm uses two variables U and V as the fitting functions. The variable U represents the number of correct prediction of τ^{A} in the last V training steps.

Attacks are based on learning by listening have in common that the opponent tries to synchronize one or more tree parity machines with sender and receiver's neural networks. Of course after the partners have reached identical weight vectors, they stop the process of synchronization, so that the number of available examples for attacks is limited. Therefore attacker's online learning is only successful if he discovers the key before the sender and receiver finish the key exchange. One way to calculate the success probability (P_E)of the attack is by:

$$P_E \sim e^{-y(L-Lo)}$$

As long as $L>L_o$. While this scaling behavior is the same for all attacks, the constant y and Lo are different for each method [18].

To provide a simple attack, an attacker has to test all possible values of weights. By K hidden neurons, K*N input neurons and boundary of weights L, this give $(2L+1)^{KN}$ possibilities , making the attack impossible with today's computer power [12].

The geometric attack is the simplest method therefore its success probability is lower than the more advanced methods, as the exponent y is large and the communicating parties can secure the key exchange protocol against this method by increasing L [18, 20].

In case of majority attack the success probability is higher; because the cooperation between the attacking networks reduces the coefficient y sender and receiver have to compensate this by further stepping up L [18, 20].

In contrast genetic attack merely increases L but the constant y does not change significantly compared to geometric attack. Therefore genetic attack is only better if L is not too large. Otherwise attacker gains most by using the majority attack [18, 20].

2.3.8. Using Neural Cryptography with Classical Cryptography

In neural cryptography, the tree parity machines are synchronized and then a secret key is generated, this secret key is used for encrypting and decrypting the secret information. A lot of research work is done on using neural and classical cryptography together where, key is generated by neural cryptography technique and encryption through classical encryption algorithms like SDES, AES, Rijindael [5, 14].

But the whole process adopted was prone to both the neural and classical cryptography security attacks. This collaboration gave security to secure transmission by eliminating the classical key distribution problems such as man-in-the-middle attack in Diffie-Hellman key exchange process, which compromised as message privacy [5]. Classical key distribution problem was tried to be solved with neural key distribution but other attacks were found to which neural key exchange was vulnerable. To achieve security of neural cryptography various other and more secure neural key exchange algorithms were proposed.

2.3.9. Other Neural key Exchange Algorithms Proposed

Security of neural synchronization is put at risk if an attacker is capable of synchronizing with any of the two parties during the training process. Therefore diminishing the probability of such a threat improves the reliability of exchanging the output bits through a public channel. The synchronization with feedback algorithm is one of the existing algorithms that enhance the security of neural cryptography. There are three new algorithms which enhance the mutual

learning process. They mainly depend on disrupting the attacker confidence in the exchanged outputs and input patterns during training [6].

a) **DO NOT TRUST MY PARTNER (DTMP) ALGORITHM**

This algorithm relies on one party sending erroneous output bits, with other party being capable of predicting and correcting this error [6].

b) SYNCHRONIZATION WITH COMMON SECRET FEEDBACK (SCSFB) ALGORITHM

In this algorithm inputs are kept partially secret and the attacker has to train its network on input patterns that are different from the training sets used by the communication parties [6].

c) **HYBRID TECHNIQUE**

The hybrid technique combines the features of DTMP and SCSFB algorithm. While the DTMP algorithm hides the real values of the mutual output bits, the SCSFB makes the input vectors partially invisible. Hence, the two mechanisms can be combined to exploit the strengths of both and produce a system that is more protected to the possible attack strategies [6].

These neural key exchange algorithms are simulated through or with the help of various neural network tools.

2.3.10. Neural network tools

Neural network tools help us to create ANN, train the designed ANN also test and run the designed ANNs. Some of the tools are discussed below in brief:

a) Matlab NeuralNetwork Toolbox

Matlab Neural Network Toolbox provides functions and apps for modelling complex nonlinear systems that are not easily modelled with a closed-form equation. Neural Network Toolbox supports supervised learning with feed forward, radial basis, and dynamic networks. It also supports unsupervised learning with self-organizing maps and competitive layers. With the

toolbox we can design, train, visualize, and simulate neural networks. To speed up training and handle large data sets, you can distribute computations and data across multicore processors, GPUs, and computer clusters using Parallel Computing Toolbox.

b) NnetLib 'C'

It consists of functions and data structures that implement neural networks for C programmers. The networks that it implements include: backpropagation, Kohonen, and the Hopfield network. It runs on Windows (DLL), DOS.

c) NeuroShell2/ Neuro Windows

NeuroShell 2 combines powerful neural network architectures, a Microsoft® Windows icon driven user interface, sophisticated utilities, and popular options to give users the ultimate neural network experimental environment. To use the system we enter data, specify the inputs and outputs, and train the network. We are then able to apply the trained network to new data and export the results to other programs.

d) Neuro Solutions

NeuroSolutions is a neural network development environment developed by NeuroDimension. It combines a modular, icon-based (component-based) network design interface with an implementation of advanced learning procedures. The software is used to design, train and deploy neural network (supervised learning and unsupervised learning) models to perform a wide variety of tasks such as data mining, classification, function approximation, multivariate regression and time-series prediction.

e) SNNS

"Stuttgart Neural Network Simulator" developed at the University of Stuttgart, Germany. With X11 interface: Graphical 2D and 3D topology editor/visualizer, training visualization, multiple patterns set handling etc. Supports various networks including backpropagation, quickprop, ART1, ART2, ARTMAP. It runs on SunOS, Solaris.

3. LITERATURE REVIEW

"A Survey on Quantum and Classical Cryptography" paper by Derrick Chait [16] highlights the importance of secure data transfer and communication through classical and quantum cryptography, their advantages and limitations. According to the survey of the author the classical key distribution systems are protected only by the limitations of the currently available computing power, with the breakthroughs in quantum cryptography classical key distribution systems are in danger. The implementation of secure quantum key distribution system practically is not guaranteed as it is in theory. A few of the problems associated with quantum key distribution system are: high bit errors during the transfer of the quantum bits, no quantum repeaters are available so the distance over which communication can happen is too short and finally sending of one photon at a time is a challenge and if it is not met the security of the system is compromised.

"Quantum Cryptography: A New Generation of Information Security System" paper by R. K. Jain et al [2] present a critical view of the working of the quantum cryptography and quantum key distribution (QKD). Also a real-world application implementation of QKD is shown where a virtual private network based on QKD is constructed; experiments demonstrate that keys can be exchanged over distance of a few tens of kilometers at rates at least of the order of a thousand bits per second. The authors conclude the paper by saying that the present quantum cryptography is still very limited in distance. These days public key systems occupy the market and every so often, classical cipher systems are broken. This would be impossible with properly implemented quantum cryptography.

"The Theory of Neural Network and Cryptography" paper written by I.Kanter et al [15] presents a relation between the theory of neural network and cryptography. A new phenomenon, namely synchronization of neural networks, is leading to a new method of exchange of secret messages. Two artificial networks being trained by the hebbian learning rule on their mutual outputs develop an antiparallel state of their synaptic weights. The synchronized weights are used to construct a key exchange protocol for the secure transmission of secret data. The complexity of the generation of the secure channel is linear with the size of the network. An attacker who

knows the protocol and all details of any transmission of the data finds it complicated to decrypt the secret message.

“Neural Cryptography for Secret Key Exchange and Encryption with AES” by A.Singh et al [5], in this paper experimental demonstration of the neural key exchange process and encryption through classical AES encryption algorithm is shown. The neural key exchange algorithm is an application of synchronization. The sender and receiver use the TPM with same structure and achieve synchronization between the machines where the number of hidden and input neurons and the synaptic weight range are public parameters whereas the initial randomly chosen weight and internal representation of the TPM are kept private. After full synchronization the modified weights are used as common secret key. Since the internal representation of the communicating parties is not known to the attacker which creates him the difficulty to eavesdrop. But there is the risk of other attack methods that could break the security of the neural cryptography which exists for all except the one time pad cryptographic algorithm.

“Design of an Efficient Neural Key Generation” written by R.M.Jogdand et al [4], this paper interacting neural networks have been calculated analytically. Through synchronization the communicating partners agree on a common secret key over a public channel. This experiment is implemented in MATLAB . Graphs are plotted which show that synchronization time will go on decreasing as the number of input increases. The eavesdropper has all information except the initial weights of the communicating partners and uses the same algorithm. Even then he does not synchronize. The randomness of the key is also achieved in the experiment where in each iteration so the opponent cannot predict the key.

“On the Improvement of Neural Cryptography Using Erroneous Transmitted Information with Error Prediction” by Ahmed M. Allam et al [6], in this paper security of neural cryptography was shown to be improved by introducing three algorithms. This paper starts with the discussion of the mutual learning method for both network and TPM. Also it summarizes the most known attacks against mutual learning. The authors have presented three algorithms: the DTMP algorithm, the SCSFB algorithm and the hybrid solution of both DTMP and SCSFB algorithms to enhance the security of neural cryptography in such a way that the attacker faces difficulties in

trusting the transmitted information on the public channel. The proposed algorithm tamper with the listening process, which is the basic mechanism the attacker, depends on to break into the system. The DTMP algorithm with error prediction does not increase the synchronization time and thus outperforms the synchronization with feedback. SCSFB is a hybrid of DTMP algorithm and synchronization with feedback scheme. It is based on the idea that the input vector becomes partially secret. Therefore the attacker cannot apply the learning rule directly without predicting the secret part of the input patterns. Similar to DTMP, SCSFB does not affect the synchronization. The DTMP and SCSFB algorithms are then combined to make use of each in hiding information. The SCSFB was shown to improve the DTMP security. The algorithm has outperformed the neural synchronization with feedback, since it does not affect the synchronization time. These strategies do not affect the attack decision and the updated weights in both hebbian and anti-hebbian learning rules, they influence only the attack decisions in random-walk learning rule. Also the DTMP with SCSFB gives better security than using DTMP alone. Therefore, only the two partners can detect the transmitted error, obtain common input vectors and hence guarantee synchronization, while the attacker faces difficulties in detecting this error.

4. MOTIVATION/JUSTIFICATION AND RELEVANCE

Security of data is a major issue nowadays and to achieve security cryptography has an important contribution. Through various classical and non-classical cryptographic techniques the data transmission is made secure. But on the other side there are many factors/drawbacks because of which the secured data could be accessed by unauthorized users. The drawbacks in classical key distribution system gave rise to new cryptographic techniques of which neural cryptography is one. The synchronization time, learning rule and security attacks are the major considerations of the neural key exchange process. There are various neural key exchange algorithms like DTMP, SCSFB, HYBRID algorithms which aim to improve the security of neural cryptography.

The DTMP algorithm is an extension to the basic neural key exchange process where erroneous output bits are sent by one partner to another who is capable of predicting and correcting the error. But till date only the random walk learning rule has given a significant effect on security

with this algorithm. There are various other learning rules through which optimized synchronization time could be achieved and could be tested for various security attacks and implemented with the help of cryptographic and neural network tools available.

5. OBJECTIVES

a) To make a detailed study of neural network, neural cryptography, neural key exchange algorithms, synchronization process, security attacks and cryptographic tools for algorithm implementation.

b) To identify and study suitable neural key exchange algorithm on the basis of security attacks.

c) To examine and analyze the limitations of the algorithm with respect to the synchronization time, learning rule and security attacks, etc.

d) To propose improvement considering the above mentioned factors in the selected algorithm.

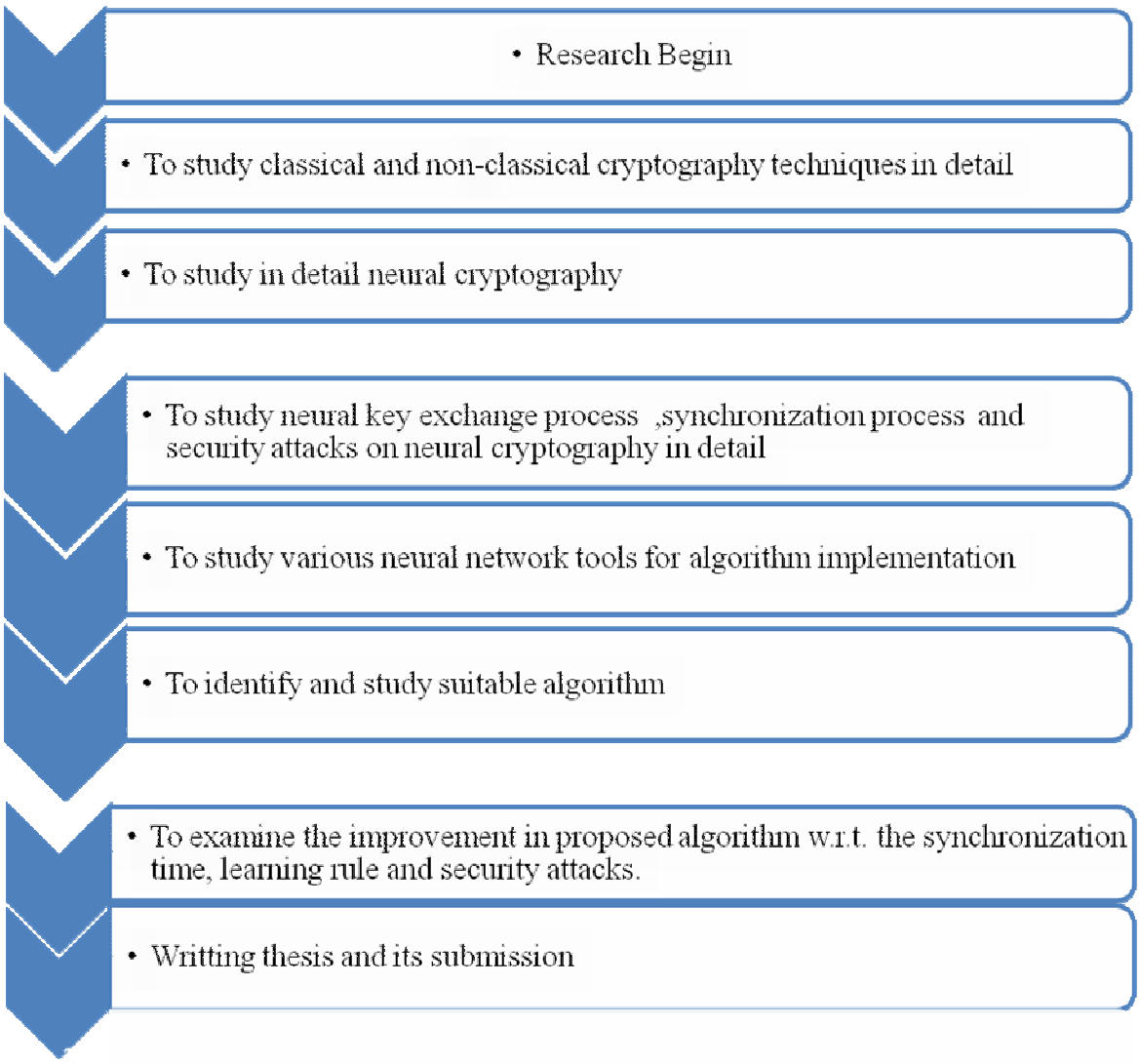
e) To develop and implement new algorithm using the suitable neural network tools or programming language with propose algorithm.

f) To analyze and compare the new algorithm with the already existing algorithms.

6. PLAN OF WORK AND METHODOLOGY

6.1. Methodology

In this research a detailed study on the security attacks and key exchange algorithm based on neural networks will be conducted. The suitable algorithm based on neural key exchange process will be selected for study. Limitation of algorithm will be examined using neural network tool like matlab neural network toolbox, SNNS, neuro solutions, etc. The algorithm will be proposed (with respective improvement) and it will be implemented using the tool. The improvement in proposed algorithm will be examined with respect to the synchronization time, learning rule and security attack.


• Research Begin
• To study classical and non-classical cryptography techniques in detail
• To study in detail neural cryptography
• To study neural key exchange process ,synchronization process and security attacks on neural cryptography in detail
• To study various neural network tools for algorithm implementation
• To identify and study suitable algorithm
• To examine the improvement in proposed algorithm w.r.t. the synchronization time, learning rule and security attacks.
• Writting thesis and its submission

6.2. Work plan

Phase 1: In depth study of classical and non classical cryptography techniques shall be done in 1-2 months.

Phase 2: 2-3 months shall be needed to study the neural cryptography in detail.

Phase 3: Next 2-3 months shall be required to study in detail the neural key exchange process, synchronization process and security attacks on neural cryptography.

Phase 4: 1-3 months will be needed to study the various neural network tools for algorithm implementation.

Phase 5: 1-2 months shall be needed to identify and study the suitable neural key exchange algorithm.

Phase 6: To examine the improvement in proposed algorithm with respect to the synchronization tine, learning rule and security attacks 3-4 months shall be required.

Phase 7: Next 4-5 months shall be required for thesis writing and its submission

7. EXPECTED OUTCOMES OF THE PROPOSED RESEARCH WORK

The research conducted would result in an improved neural key exchange algorithm with lesser vulnerabilities and greater securities

8. PLACE OF WORK AND FACILITIES AVAILABLE

The experimental setup for the tool development shall be set up in the Information Technology Lab in THE IIS UNIVERSITY Campus and all the resources and facilities required for doing the research are available

9. REFERENCES

1) C.H. Bennett and G.Brassard, "An Update on Quantum Cryptography", G.R.Balkely and D.Chaum (Eds):Advances in Cryptology- CRYPTO'84, LNCS 196, Springer- Verlag Berlin Heidelberg, pp. 475-480,1985.

2) R. K. Jain, K. Hiran and G. Paliwal, "Quantum Cryptography:A New Generation Of Information Security System", *Proceedings of International Journal of Computers and Distributed Systems*, ISSN:2278-5183, Vol.No. 2,Issue 1, pp. 42-45, December 2012.

3) V. Sharma, S. Rai and A. Dev, "A Comprehensive Study Of Artificial Neural Networks", *International Journal Of Advanced Research in Computer Science and Software Engineering*, Vol. 2, Issue 10, ISSN: 2277 128X , pp: 278-284, October 2012

4) R. M. Jogdand and S. S. Bisalpur, "Design Of An Efficient Neural Key Generation", *International Journal Of Artificial Intelligence & Applications (IJAIA*), Vol.2 , No. 1, January 2011.

5) A. Singh and A. Nandal, "Neural Cryptography for Secret Key Exchange and Encryption with AES*"*, *International Journal Of Advanced Research in Computer Science and Software Engineering*, Vol. 3, Issue 5,ISSN: 2277 128X , pp: 376-381, May 2013

6) A. M. Allam and H. M. Abbas, "On The Improvement Of Neural Cryptography Using Erroneous Transmitted Information with Error Prediction", *IEEE Transactions on Neural Networks* , Vol. 21, Issue 12, ISSN 1045-9227, December 2010.

7) B. M. Wilamowski, "Neural Network Architectures and Learning", *Industrial Electronics Magazine, IEEE,* Vol. 3, Issue 4, ISSN 1932-4529, pp: 56-63, December 2009.

8) A. M. Allam, H. M. Abbas and M. W. El-Kharashi, " Security Analysis of Neural Cryptography Implementation ", *IEEE Pacific Rim Conference on Communications, Computers and Signal Processing (PACRIM*), ISSN 1555-5798, pp: 195-199, August 2013.

9) R. Mislovaty, E. Klein and W. Kinzel, " Security Of Neural Cryptography ", *Proceedings of 11th IEEE International Conference on Electronics, Circuits and Systems (ICECS)*, ISBN 0-7803-8715-5, pp: 219-221, December 2004.

10) A. M. Allam and H. M. Abbas, "Improved Security Of Neural Cryptography Using Don't-Trust-My-Partner and Error Prediction", *Proceedings of IEEE International Joint Conference on Neural Network Atlanta, Georgia,USA*, ISSN 1098-7576, pp: 121-127, June 2009.

11) A. Kilmov, A. Mityagin and A. Shamir, "Analysis Of Neural Cryptography", *Advances in Cryptography ASIACRYPT*, *Springer-Verlag Berlin Heidelberg,* pp: 288-298, 2002.

12) V. Soni, S. Tanwar, "Secure Key Exchange using Neural Network ", *International Journal Of Engineering Sciences & Research Technology (IJESRT)*, ISSN 2277-9655, pp:1707-1709, March 2014.

13) P. Revankar, W.Z. Gandhare, and D. Rathore, "Private Inputs To Tree Parity Machine", *International Journal Of Computer Theory and Engineering,* Vol. 2, Issue 4, pp:665-669, August 2010.

14) K. Alallayah, M. Amin, W. AbdElwahed, A. Alhamami, "Applying Neural Networks for Simplified Data Encryption (SDES) Cipher System Cryptanalysis ", *The International Arab Journal of Information Technology*, Vol 9, Issue 2, pp:163-169, March 2012.

15) I. Kanter and W. Kinzel "The Theory of Neural Networks and Cryptography", *Quantum Computers And Computing*, Vol 5, Issue 5, pp:130-140, 2005.

16) D. Chait, "A Survey on Quantum and Classical Cryptography" , *Student Paper E-Journal, The Consortium for Computing Sciences in College: South Central Region,* Vol. 1, 2008.

17) E. Volna, M. Kotyrba, V. Kocian and M. Janosek, "Cryptography Based on Neural Network", *26th Europen Conference on Modelling and Simulation (ECMS)*, ISBN 978-0-9564944-4-3, 2012.

18) A. Ruttor, W. Kinzel, R. Naeh and I. Kanter, "Genetic Attack on Neural Cryptography", *Physical Review E, American Physical Society,* Vol. 73, Issue 3, pp: 036121- 036129, March 2006.

19) K. Nahar, "Artificial Neural Network", *COMPUSOFT, An international journal of advanced computer technology*, Vol. 1, Issue 2, ISSN: 2320-0790, pp: 25-27, December 2012.

20) A. Ruttor, "Neural Synchronization and Cryptography", *PhD. Thesis, Bayerische Julius-Maximilians- University, Wurzburg,* 2006.